International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 12 (2017) pp. 3282-3286
© Research India Publications. http://www.ripublication.com

Design and Implementation of Tizen IVI-based MobileSecond Platform Architecture & its Applications

Joongjin Kook[1]
Korea Electronics Technology Institute, Seoul, Korea.

Junghoon Shin[2]
School of Software, Soongsil University
Seoul 06978, Korea.

[1]ORCID: 0000-0002-0033-388X

Abstract

While the conventional vehicle's Head-Units played very basic roles related to vehicles such as the control of HVAC (Heating Ventilation and Air Conditioning) and the radio reception, they have evolved to serve as an interface between the car and the driver with the advent of the concept of Connected Car and the development of ICT technology. IVI(In Vehicle Infotainment) device, which is Connected Car's Head-Unit, provides various functions such as AV, navigation, information related to vehicle's parts (ex; air pressure, oil gauge) and payment as well as Head-Unit's unique function. MobileSecond Platform architecture is designed to provide more powerful and diverse convergence services for vehicles and drivers by applying technologies of Connected Car and ICT Convergence in various ways. MobileSecond platform is implemented by applying Tizen IVI to IVI HW platform. In this paper, the fundamental structure of MobileSecond Platform and the vehicle control framework, which is the key element of Connected Car, are developed as native applications and Web applications. This paper also introduces the research results of vehicle control methods based on IVI devices and the service models based on MobileSecond Platform.

Keyword: MobileSecond, ICT Car, Connected Car, In-Vehicle-Infotainment

INTRODUCTION

The concept of Mobile First[1], proposed by Luke Wroblewski in 2009, suggests that all of the functions are increasingly focused on mobile devices due to the proliferation of mobile devices. The tasks, previously performed by PCs, have carried out by mobile devices, which means the mobile-oriented world has arrived. Mobile Second Strategy, on the other hand, aims to combine with new industries and create new services by expanding the work done by mobile devices to other devices.

ICT(Information and Communication Technology) convergence technology is increasingly being added to automobiles, transforming automobiles into new IT-intensive platforms, not just a transportation. MobileSecond platform linked to ICT Car, which is combined with ICT convergence technology, is an integrated platform to provide various services related to the vehicle, operation and driver by connecting to a vehicle and mobile devices. It also enables new variety of services such as smart driving service, smart care/ self-maintenance service and entertainment service to be created.

The existing Head-Units of the vehicle provided piecemeal functions such as the control of HAVC (Heating Ventilation and Air Conditioning) and the radio reception, but with the emergence of the concept of Connected Car[2], the Head-Units have evolved into IVI (In Vehicle Infotainment) devices, providing various functions such as AV, navigation, information related to vehicle's parts (ex: air pressure, oil gauge, etc.) and payment. MobileSecond platform is designed and developed by further expanding the concept of Connected Car and evolving the Mobile First strategy simultaneously. It is possible that monitoring and controlling the information of IVI device-based vehicle, linking to mobile devices and cloud, and discovering new service models through this.

In this paper, the basic structure of MobileSecond platform is introduced and the vehicle control framework needed to implement the most basic service, Smart Driving Service and the development methods of IVI application based on it are examined. It is expected that this paper helps the control methods of the vehicle based on the MobileSecond platform and new types of convergence services to be developed.

In Vehicle Infotainment Device

Beyond the simple network connection, Connected Car connects the car and mobile devices, connects to other vehicles, and connects to various services such as driving, accident, maintenance, entertainment. By connecting to homes, offices, and various infrastructures, it transforms automobiles into the new form of IT-intensive technologies, not just a transportation. Connected Car platform should include components for Infotainment, Safety, Diagnostics, Efficiency, Navigation, and Payment[3].

IVI(In-Vehicle-Infotainment) mainly focused on the roles of AV equipment and navigation function. By combining with HAVC and receiving various sensor information from ECU, it has also provides the condition of the vehicle for the driver. It also responds to driver's requests based on voice recognition, and it also enables the smart phone's unique service to be used in IVI by linking to a smart phone. IVI is an interface between the car and the driver. QNX's Car2[4], MS's Embedded Windows Automotive 7[5], MontaVista's Automotive Technology Platform (ATP)[6], Samsung's Tizen IVI[7] and Wind River Automotive IVI Platform[8] are typical SW platforms providing such services. These IVI platforms are customized in their own forms by the manufacturer's vehicle models or compatible with the GENIVI standard.

Vehicle Selective Gateway

Data generated by the main components and sensors of the vehicle are collected and controlled through ECUs(Electronic Control Unit), and ECUs are configured by various modules of the vehicle. Each ECU is equipped with software components(SW-C) to be processed, and ECUs construct the network through CAN BUS as shown in Figure 1. For example, braking, chassis, steering, and powertrain ECUs manage their own sensors and actuators separately[9].

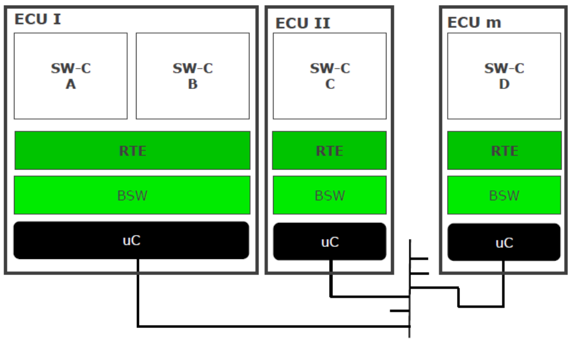

Figure 1. Layered Architecture of ECUs

For SW standards, there are OSEK and AUTOSAR; OSEK consists of OSEK OS and OSEK COM modules that are responsible for ECU-based task management and communication, and AUTOSAR, which is a middleware standard, enables HW to be independently executed by providing Micro Controller of ECU and the abstraction layer of ECUs and by minimizing HW dependency of the application SW. Figure 2 shows the AUTOSAR architecture

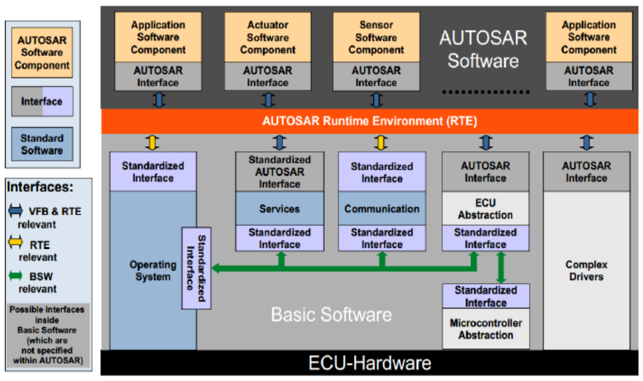
AUTOSAR Software Component
Interface
Standard Software
Interfaces:
VFB & RTE relevant
RTE relevant
BSW relevant
Possible interfaces inside Basic Software (which are not specified within AUTOSAR)
Application Software Component
Actuator Software Component
Sensor Software Component
AUTOSAR Software
AUTOSAR Interface
AUTOSAR Runtime Environment (RTE)
Standardized Interface
Standardized AUTOSAR Interface
Services
Communication
ECU Abstraction
Operating System
Standardized Interface
Complex Drivers
Basic Software
Microcontroller Abstraction
ECU-Hardware

Figure 2. AUTOSAR Architecture

An interface between the ECU and the IVI is required to monitor the condition of the vehicle via IVI and to forward commands via the touch screen or voice. The structure of the ECU is shown in Figure 3, which represents Freescale's Body Control Module. ECU provides digital/ analog I/O to control sensors and actuators and consists of CAN, LIN and MOST as means for network of ECU.

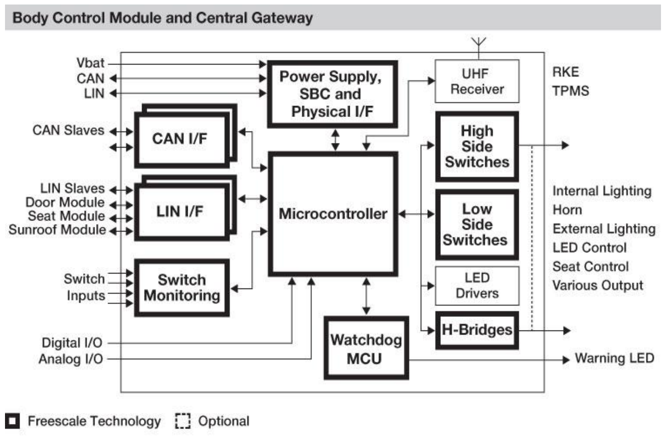

Figure 3. Freescale Body Control Module and Central Gateway

The VSG serves as a gateway between the vehicle and the mobile second device, and can be implemented as a software embedded in the ECU or as a separate hardware. Automobile manufacturers typically implement ECU-based SWs, but in this paper, VSG based on FreeScale's MC9S08DZ60 is developed for experiments. The gateway architecture based on MC9S08DZ60 is shown in Figure 4, which can obtain and control the required signals by being connected to the ECU of the vehicle via CAN.

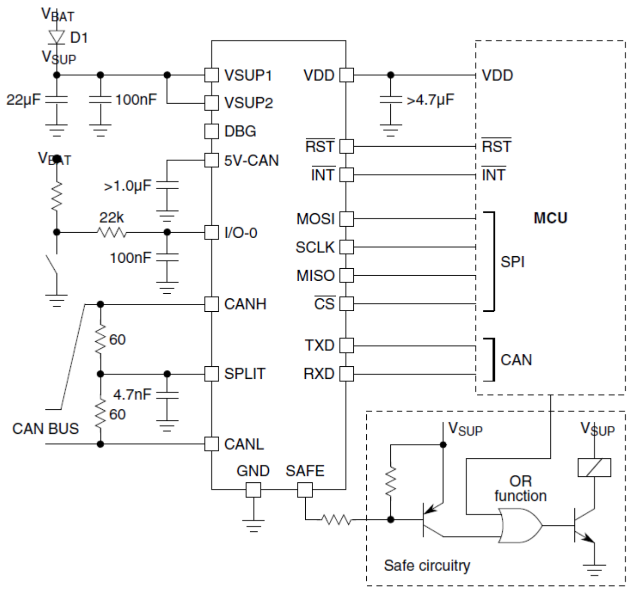

Figure 4. Vehicle Selective Gateway Block Diagram

The VSG implements the API to provide the internal information of the vehicle for the IVI and to receive the control commands through the IVI, as follows:

Table 1. VSG APIs

API Function	Description
getStartVehicleStatus( )	Starting status of the vehicle

getDriverDoorOpen( )
getAssistDoorOpen( )
	driver/passenger locked/open

getFuelLevel( )	Fuel quantity status
getEngineRPM( )	Engine RPM information
getTotalDistance( )	Total mileage information
getDriveSpeed( )	Speed of the vehicle
getAccCount( )	No. of rapid acceleration
getRedCount( )	No. of rapid deceleration
getPBreak( )	Parking gear status
getAirbagStatus( )	Airbag deployment status
getCheckEngine( )	Engine abnormal status
getVehicleID( )	Plate number information
getSideBrakeStatus( )	Parking brake status

Like Figure 5, VSG and IVI communicate with each other via the API defined above, and they enable the vehicle to be controlled outside by being linked to the cloud.

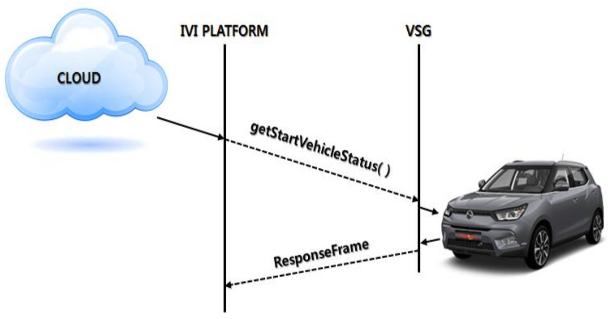

Figure 5. VSG-to-IVI Communication

MobileSecond Platform SW Architecture

MobileSecond defines the types of services provided to drivers during the actual driving by expanding and improving the concept of Connected Car and details the necessary technical elements to realize them specifically.

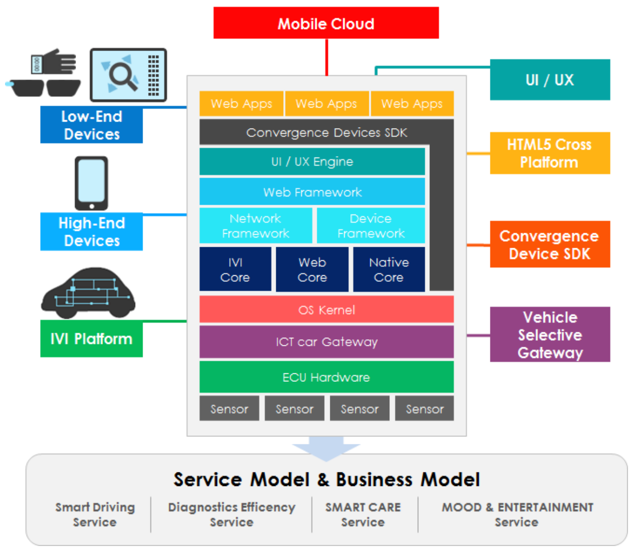

Figure 6. MobileSecond Platform Overall Architecture

Figure 6 shows SW Architecture of MobileSecond Platform. The ECU, which manages the sensors and actuators related to various functions inside the vehicle, is connected to the ICT Car Gateway, the VSG via the CAN, and the VSG is connected to the IVI to serve as an interface between the vehicle and the driver. IVI consists of IVI core, web core, and Native core components to perform the Head-up Unit's original functions, and provides Convergence Devices SDK to enable the development of Web-app and Native App. It provides the vehicle status information for the driver, responds to the driver's requests, and can interwork with other external services by linking to the cloud.

The basic services linked to MobileSecond Platform are Smart Driving Service, Diagnostics Efficiency, Smart Care Service and Mood & Entertainment Service. These enable the driver to easily cope with various situations that may occur during driving and provide a pleasant driving environment.

In this paper, Nexcom's VTC 1010 is used as an IVI device to connect to VSG. The main features of VTC 1010 are as follows:

- Intel® Atom™ processor E3827, 1.75GHz
- Dual SIM cards + dual WWAN modules support
- Wide operating temperature -30°C ~ 70°C
- Built-in CAN 2.0B. Optional CAN/OBDII module (CAN Bus 2.0B or OBDII SAE J1939)
- 4 x mini-PCIe socket rich expansion capability
- Wake on RTC/SMS via WWAN module
- Voice communication via WWAN module
- Compliant with MIL-STD-810G

- Built-in U-blox M8N GPS, optional dead reckoning support

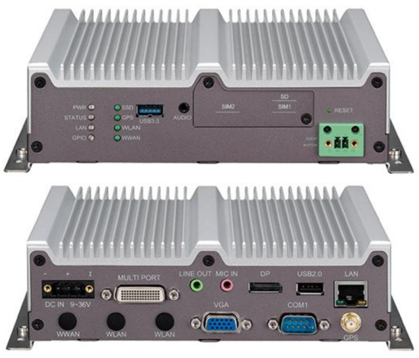

Figure 7. Nexcom VTC 1010

Figure 8 shows the state of Tizen IVI ported to VTC 1010.

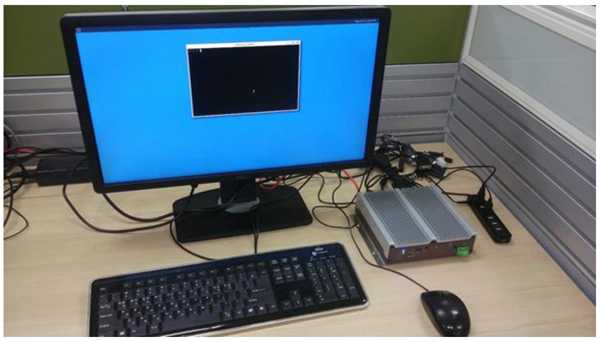

Figure 8. Tizen IVI ported on VTC 1010

VTC 1010 supports one of the IVI SW platforms, Samsung's Tizen IVI 2.0, and Tizen IVI includes both a framework for native applications and a framework for web applications.

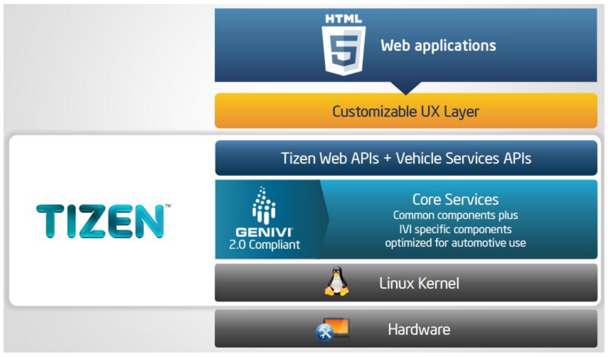

Figure 9. Tizen IVI Architecture

IVI's SW uses both Native Framework and Web Framework, and implements Native Framework-based UART library and calls the VSG API. In order to implement UI and services, it is implemented as Web-App based on Crosswalk which is a cross framework. The UART library to call the vehicle control API of the VSG defines a byte array corresponding to the VSG's command packets as follows.

Table 2. ECU Command Packets

```
uint8_t get_driver_seatbelt_status[] =
{0xAA, 0x41, 0x10, 0x01, 0x00, 0x00, 0x00, 0x55};
uint8_t get_start_vehicle_status[] =
{0xAA, 0x41, 0x13, 0xFF, 0x00, 0x00, 0x00, 0x55};
uint8_t set_start_vehicle[] =
 {0xAA, 0x51, 0x13, 0xFF, 0xFF, 0x00, 0x00, 0x55};
uint8_t set_stop_vehicle[] =
{0xAA, 0x51, 0x13, 0xFF, 0x00, 0x00, 0x00, 0x55};
uint8_t get_driver_temperature_c[] =
{0xAA, 0x41, 0x21, 0x01, 0x00, 0x00, 0x00, 0x55};
uint8_t set_driver_temperature_c[] =
{0xAA, 0x51, 0x21, 0x01, 0x80, 0x80, 0x00, 0x55};
uint8_t get_passenger_temperature_c[] =
{0xAA, 0x41, 0x21, 0x10, 0x00, 0x00, 0x00, 0x55};
...
```

The commands in Table 2 correspond to the vehicle model of the manufacturer we used for the experiment, and in this experiment, Hyundai Motors' SONATA is used.

The control commands to be transferred to the VSG can be used in IVI applications by wrapping them with API functions in Table 3 and building them into libraries. Examples of Node.js application based on the UART library are as follows:

Table 3. VSG APIs & Node.js Example

```
VSG UART Module Functions
1. Load serial module
var my = require('./build/Release/serial');
2. getDriverSeatBeltStatus()
--> ret: 1 (ON)
--> ret: 0 (OFF)
console.log(my.getDriverSeatBeltStatus());
3. getStartVehicleStatus()
--> ret: 1 (ON)
--> ret: 0 (OFF)
console.log(my.getStartVehicleStatus());
```

```
4. setStartVehicle(unsigned char onoff)

--> onoff: 1 (ON)

--> onoff: 0 (OFF)

console.log(my.setStartVehicle(onoff));

...
```

The reason for using Node.js in IVI's VSG control application is to make the API call of VSG easier in mobile devices and cloud environment in the future. When server functions are added to VSG application by additionally using WebSocket or Socket.IO, the remote control of the vehicle can be performed.

CONCLUSION

MobileSecond platform is an architecture for further expanding Connected Car. It provides drivers with various services such as Smart Driving, Smart Care, Diagnostics Efficiency and Mood & Entertainment that may occur during driving as well as various information on the vehicle. In this paper, in relation to Smart Driving Service, the methods for IVI responsible for an interface between the car and the driver, to include these functions are studied and implemented by studying a framework related to the control of the vehicle that the MobileSecond platform should provide. It is impossible to apply the platform we developed to all the finished vehicles because it is required to obtain all of the information necessary for the control of the vehicle from the manufacturers. However, it is expected to serve as a reference architecture for automobile manufactures to make Connected Car environment more evolved and provide various services.

ACKNOWLEDGMENT

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIP) (NRF-2016R1C1B1010984)

REFERENCES

[1] Luke Wroblewski, Mobile First, https:// www.lukew.com/ff/entry.asp?933

[2] McKinsey & Company, “Connected car, automotive value chain unbound,” Apr. 2017.

[3] Connected Car, http://www.autoconnectedcar.com /definition-of-connected-car-what-is-the-connected-car-defined/

[4] QNX Car 2, http://www.qnx.com/content/qnx/en /products/qnxcar/index.html

[5] Microsoft Windows Embedded Automotive 7, https://www.microsoft.com/windowsembedded/en-us/windows-embedded-automotive-7.aspx

[6] MontaVista, Automotive Technology Platform, http://www.mvista.com/press_release_detail1.php?fid=news/2011/MontaVista_ATP_Declared_GENIVI_Compliant.html&d=530

[7] Samsung, Tizen IVI, https://wiki.tizen.org/IVI

[8] Wind River, Automotive IVI Platform, https://www.windriver.com/news/press/pr.html?ID=12843

[9] Rudolf Grave, Alexander Much, “Evolving Needs for Software Systems – Demonstrated,” VDA Automotive SYS 2016, Jul., 2016.